

Cont  
Sub A32

(Ultra 2/Ultra) SCSI controller. The wide SCSI controller 610 in turn drives one or more data storage devices 612.

The above described software can be implemented in a high level procedural or object-oriented programming language to operate on a dedicated or embedded system.

5 However, the programs can be implemented in assembly or machine language, if desired. In any case, the language may be a compiled or interpreted language.

Each such computer program can be stored on a storage medium or device (e.g., CD-ROM, hard disk or magnetic diskette) that is readable by a general or special purpose programmable computer for configuring and operating the computer when the storage medium  
10 or device is read by the computer to perform the procedures described. The system also may be implemented as a computer-readable storage medium, configured with a computer program, where the storage medium so configured causes a computer to operate in a specific and predefined manner.

While the invention has been shown and described with reference to an embodiment  
15 thereof, those skilled in the art will understand that the above and other changes in form and detail may be made without departing from the spirit and scope of the following claims.

Other embodiments are within the scope of the following claims.

What is claimed is:

1 1. An operating system, comprising:  
2 a non-preemptive microkernel executing one or more processes in accordance with a  
3 predetermined priority; and  
4 one or more kernels adapted to be executed as one or more processes by the non-  
5 preemptive microkernel.

1 2. The operating system of claim 1, wherein one of the kernels execute an operating system  
2 as a dependent process.

Sub A33  
2 3. The operating system of claim 2, wherein the operating system is a time-sliced operating  
system or a microkernel.

1 4. The operating system of claim 2, wherein the operating system is Unix.

1 5. The operating system of claim 1, wherein each process has its own stack.

1 6. The operating system of claim 1, wherein the processes communicate using one or more  
2 messages.

1 7. The operating system of claim 1, wherein each process has a unique process identifier  
2 (PID).

1 8. The operating system of claim 7, further comprising a mailbox coupled to a plurality of  
2 processes to service messages sent to a single PID.

1 9. The operating system of claim 1, wherein the processes never terminate.

1 10. The operating system of claim 1, wherein one of the kernels is a microkernel.

1 11. A method for operating a computer system, comprising:  
2 managing one or more processes with a non-preemptive microkernel, the microkernel  
3 running the one or more processes in accordance with a predetermined priority; and  
4 executing one or more kernels as one or more processes managed by the non-preemptive  
5 microkernel.

1 12. The method of claim 11, further comprising executing an operating system in one of the  
2 microkernels as a dependent process.

13. The method of claim 12, wherein the operating system is a time-sliced operating system  
or a microkernel.

1 14. The method of claim 12, wherein the operating system is Unix.

1 15. The method of claim 11, wherein each process has its own stack.

1 16. The method of claim 11, further comprising performing inter-process communication  
2 using one or more messages.

1 17. The method of claim 11, wherein each process has a unique process identifier (PID).

1 18. The operating system of claim 17, further comprising servicing messages sent to a single  
2 PID by a plurality of processes using a mailbox.

1 19. The method of claim 11, further comprising executing the processes without termination.

1 20. The method of claim 11, further comprising executing a microkernel in one of the  
2 kernels.

1 21. A computer system, comprising:  
2 means for managing one or more processes with a non-preemptive microkernel, the  
3 microkernel running the one or more processes in accordance with a predetermined priority; and  
4 means for executing one or more kernels as one or more processes managed by the non-  
5 preemptive microkernel.

1 22. The system of claim 21, further comprising means for executing an operating system in  
2 one of the microkernels.

1 23. The method of claim 12, wherein the operating system is a time-sliced operating system.

1 24. The method of claim 12, wherein the operating system is Unix.

1 25. The system of claim 21, wherein each process has its own stack.

1 26. The system of claim 21, further comprising means for performing inter-process  
2 communication using one or more messages.

1 27. The system of claim 21, wherein each process has a unique process identifier (PID).

1 28. The operating system of claim 17, further comprising means for servicing messages sent  
2 to a single PID by a plurality of processes using a mailbox.

1 29. The system of claim 21, further comprising means for executing the processes without  
2 termination.

1 30. The system of claim 21, further comprising means for executing a microkernel in one of  
2 the kernels.

1 31. A computer, comprising:

an interconnect bus;  
one or more processors coupled to the interconnect bus and adapted to be configured for server specific functionalities including network processing, file processing, storage processing and application processing;  
a configuration processor coupled to the interconnect bus and to the processors, the configuration processor dynamically assigning processor functionalities upon request;  
one or more data storage devices coupled to the processors and managed by a file system;  
a non-preemptive microkernel executing one or more processes in accordance with a predetermined priority; and  
one or more kernels adapted to be executed as one or more processes by the non-preemptive microkernel.

32. The computer of claim 31, wherein the microkernel executes an operating system as a dependent process.

33. The computer of claim 31, wherein the microkernel executes a network switch operating system as a dependent process